US 20140143723A1

(19) United States

(12) Patent Application Publication

(10) Pub. No.: US 2014/0143723 A1

(43) Pub. Date: May 22, 2014

Ording et al.

(54) **USER INTERFACE FOR PROVIDING CONSOLIDATION AND ACCESS**

(71) Applicant: **Apple Inc.**, Cupertino, CA (US)

(72) Inventors: **Bas Ording**, San Francisco, CA (US); **Steven P. Jobs**, Palo Alto, CA (US); **Donald J. Lindsay**, Mountain View, CA (US)

(73) Assignee: **Apple Inc.**, Cupertino, CA (US)

(21) Appl. No.: **14/164,922**

(22) Filed: **Jan. 27, 2014**

Related U.S. Application Data

(63) Continuation of application No. 13/252,227, filed on Oct. 4, 2011, now Pat. No. 8,640,044, Continuation of application No. 13/252,232, filed on Oct. 4, 2011, now Pat. No. 8,640,045, said application No. 13/252,227 is a continuation of application No. 12/410,354, filed on Mar. 24, 2009, now Pat. No. 8,032,843, said application No. 13/252,232 is a continuation of application No. 12/410,354, filed on Mar. 24, 2009, now Pat. No. 8,032,843, which is a continuation of application No. 11/892,153, filed on Aug. 20, 2007, now Pat. No. 7,526,738, which is a continuation of application No. 09/467,074, filed on Dec. 20, 1999, now Pat. No. 7,434,177.

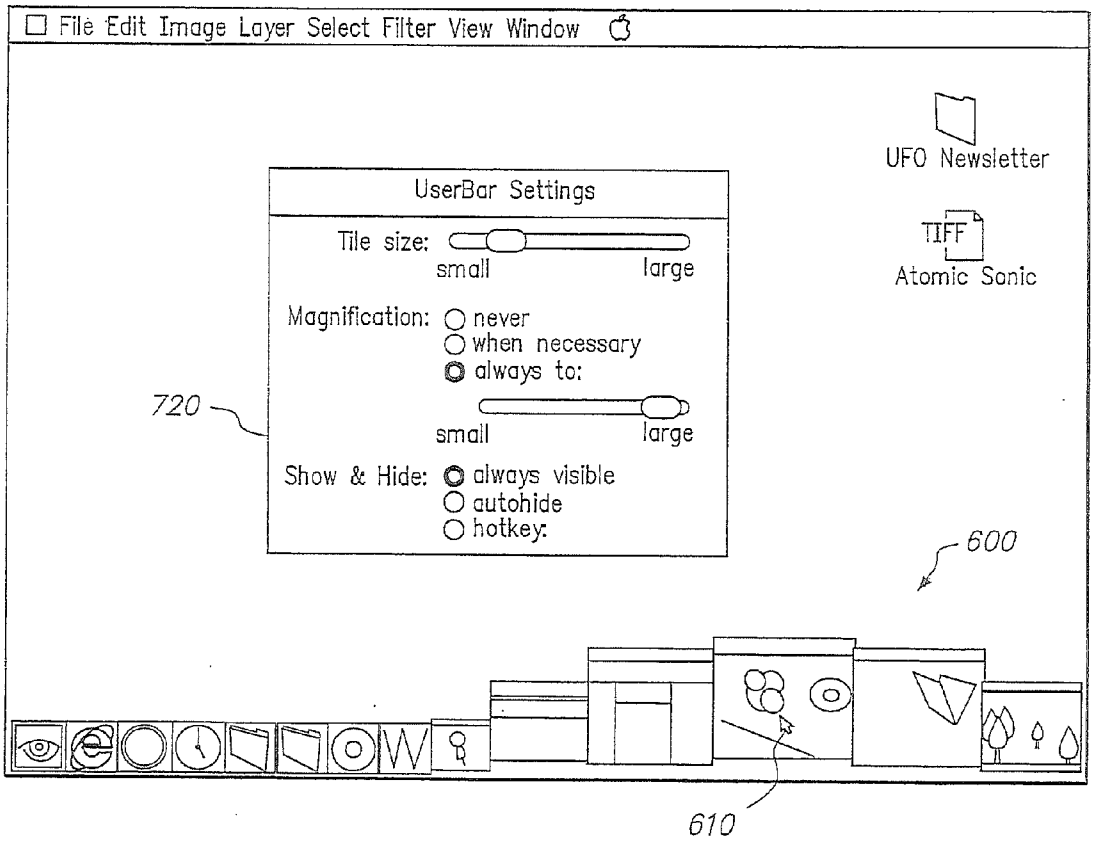
Publication Classification

(51) **Int. Cl.**
G06F 3/0484 (2006.01)
G06F 3/0482 (2006.01)

(52) **U.S. Cl.**
CPC ***G06F 3/04842*** (2013.01); ***G06F 3/0482*** (2013.01)
USPC **715/823**

(57) **ABSTRACT**

Methods and systems for providing graphical user interfaces are described. To provide greater access and consolidation to frequently used items in the graphical user interface, a userbar is established which includes a plurality of item representations. To permit a greater number of items to reside in the userbar, a magnification function can be provided which magnifies items within the userbar when they are proximate the cursor associated with the graphical user interface.



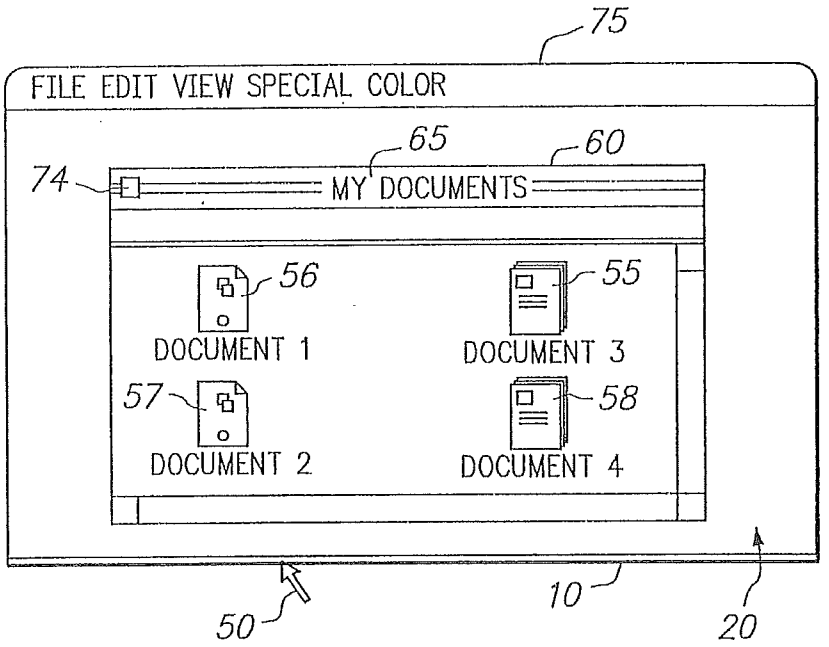


FIG. 1A (PRIOR ART)

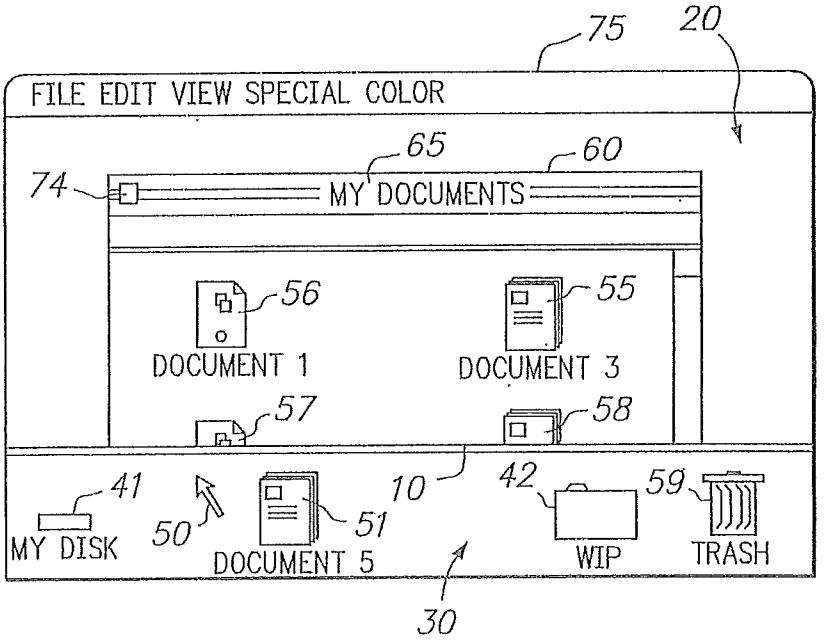


FIG. 1B (PRIOR ART)

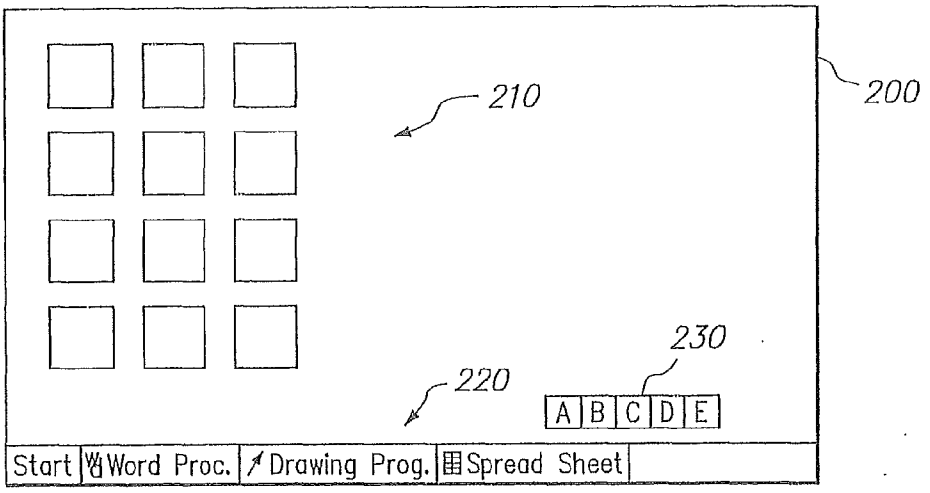


FIG. 2 (PRIOR ART)

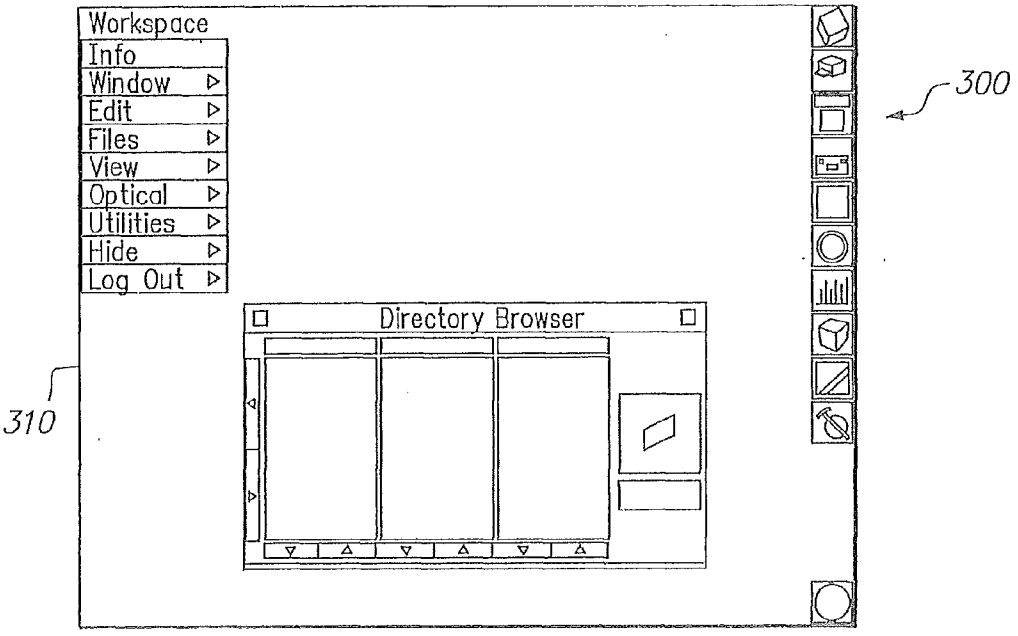


FIG. 3 (PRIOR ART)

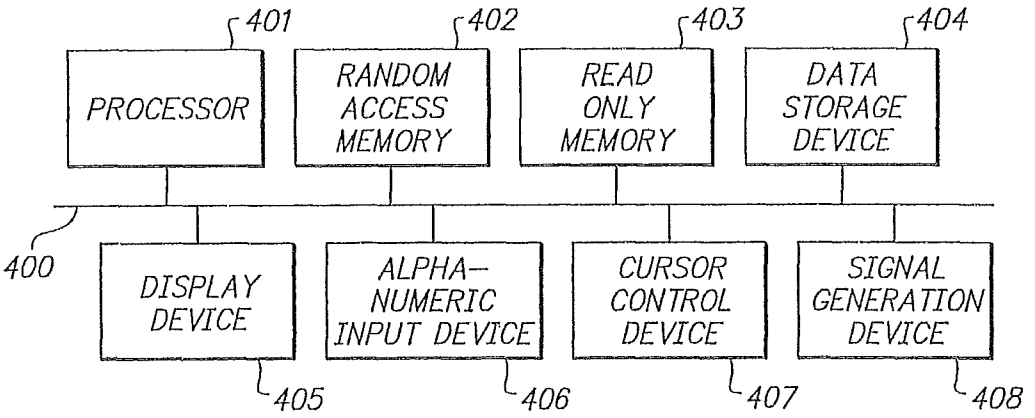

401
402
403
404
PROCESSOR
RANDOM ACCESS MEMORY
READ ONLY MEMORY
DATA STORAGE DEVICE
400
DISPLAY DEVICE
ALPHA-NUMERIC INPUT DEVICE
CURSOR CONTROL DEVICE
SIGNAL GENERATION DEVICE
405
406
407
408

FIG. 4

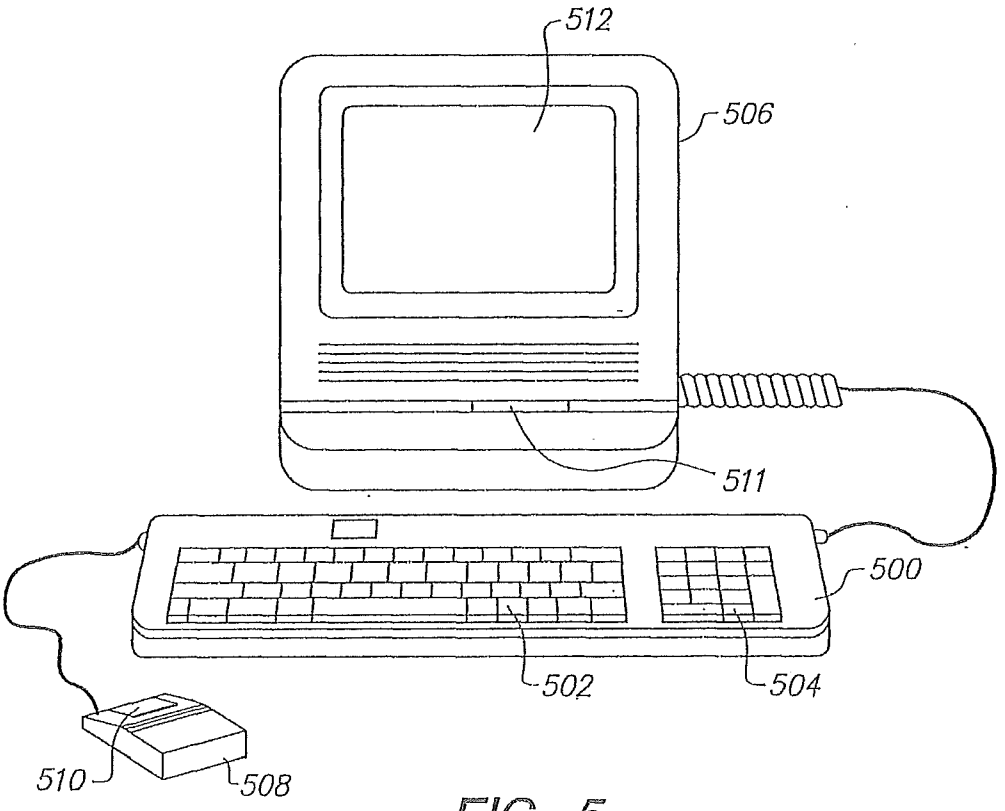

512
506
511
500
502
504
510
508

FIG. 5

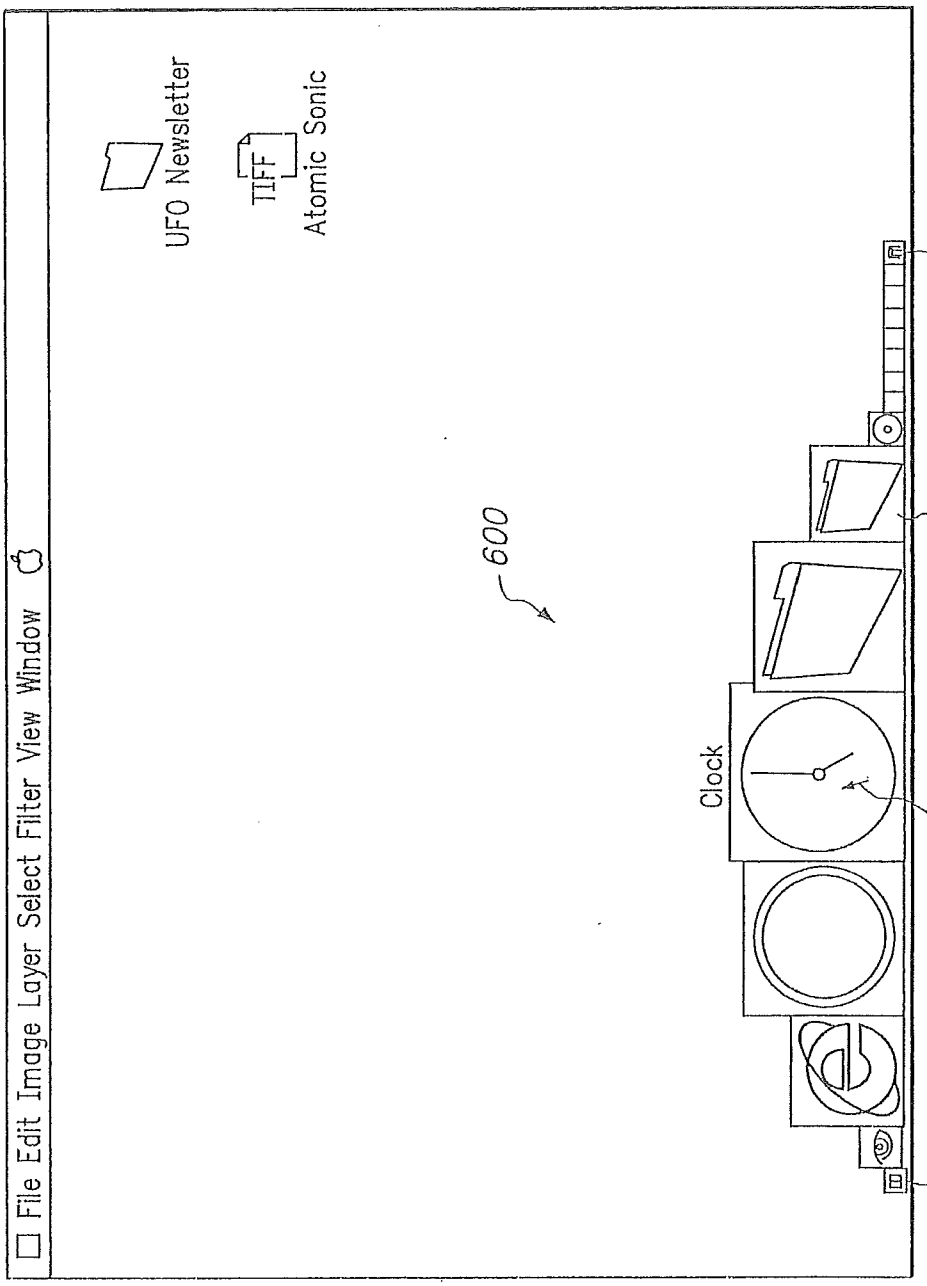

File Edit Image Layer Select Filter View Window
UFO Newsletter
TIFF
Atomic Sonic
600
Clock
610
620
630
640

FIG. 6

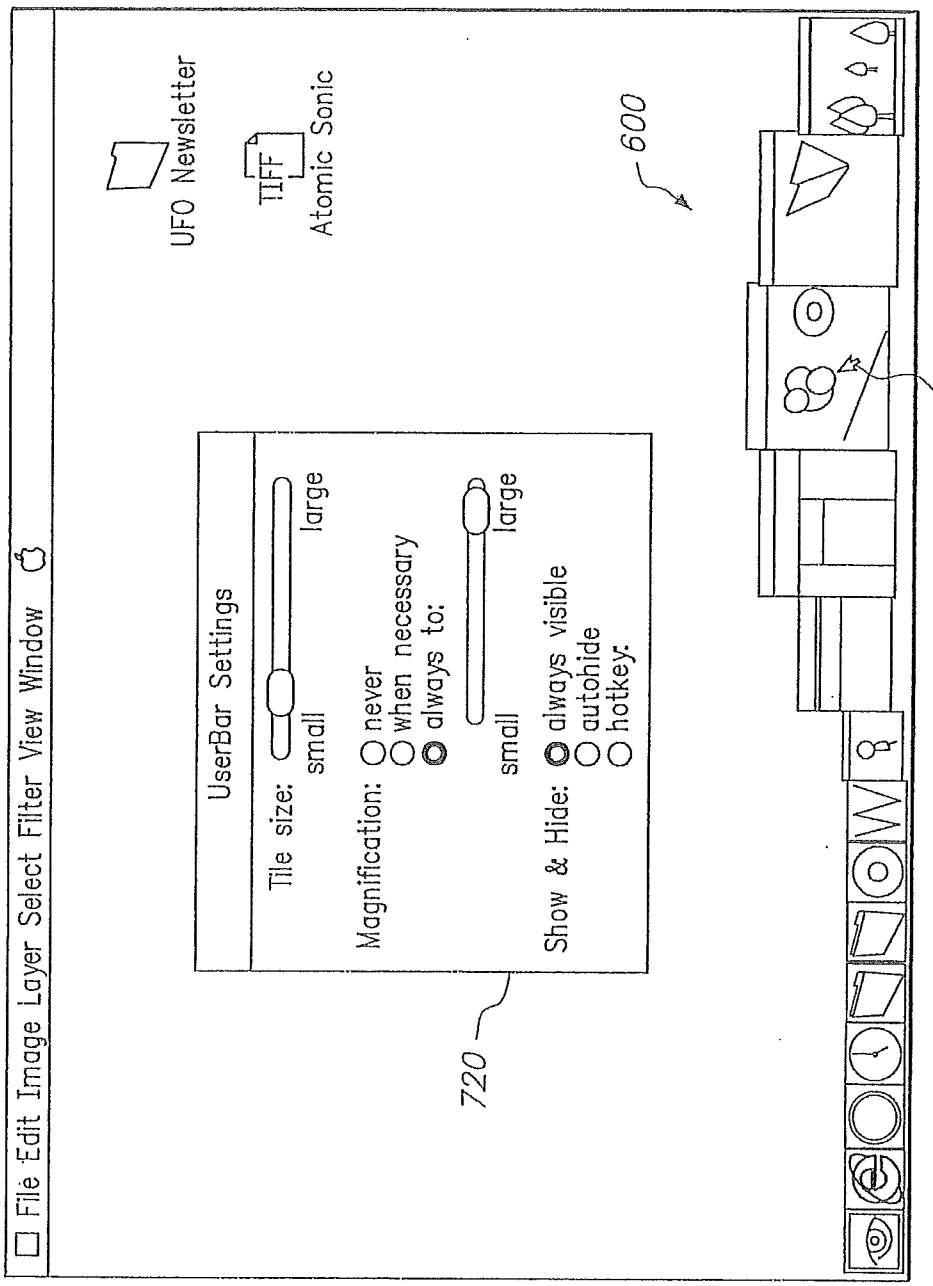

File Edit Image Layer Select Filter View Window
UFO Newsletter
TIFF
Atomic Sonic
600
UserBar Settings
Tile size:
small
large
Magnification:
never
when necessary
always to:
small
large
Show & Hide:
always visible
autohide
hotkey:
720
610

FIG. 7

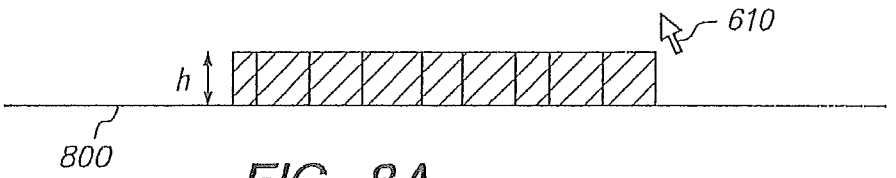


FIG. 8A

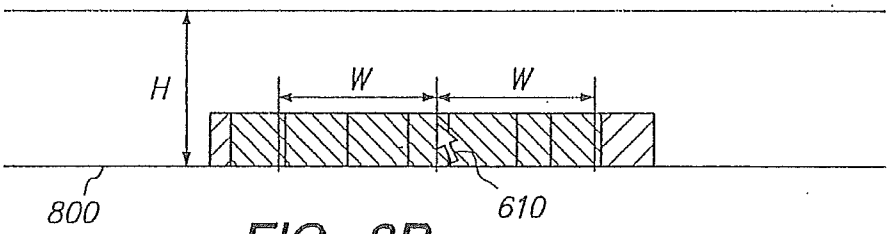


FIG. 8B

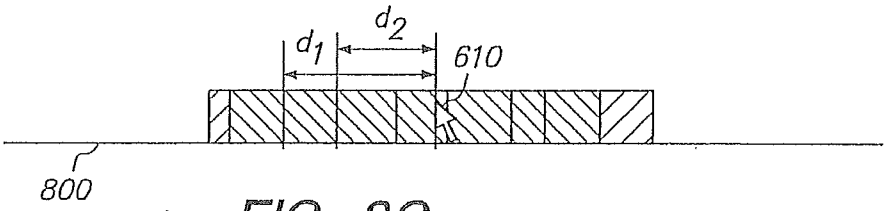


FIG. 8C

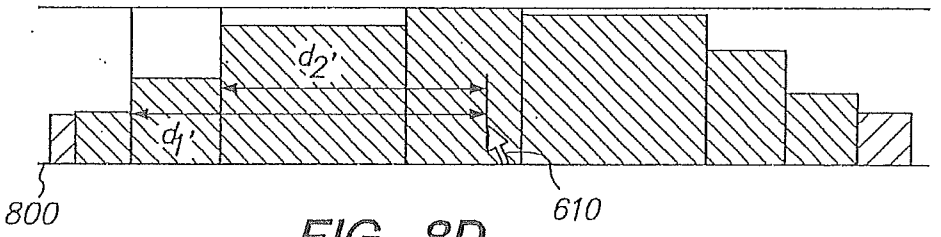


FIG. 8D

USER INTERFACE FOR PROVIDING CONSOLIDATION AND ACCESS

CROSS REFERENCE TO RELATED APPLICATIONS

[0001] This is a continuation of application Ser. No. 12/410,354, filed Mar. 24, 2009, now U.S. Pat. No. 8,032,843, issued Oct. 4, 2011, which is a continuation of application Ser. No. 11/892,153, filed Aug. 20, 2007, now U.S. Pat. No. 7,526,738, issued Apr. 28, 2009, which is a continuation of application Ser. No. 09/467,074, filed Dec. 20, 1999, now U.S. Pat. No. 7,434,177, issued Oct. 7, 2008.

BACKGROUND

[0002] The present invention relates generally to graphical user interfaces for computer systems. More particularly, the present invention relates to systems and methods for interfacing applications and operating systems which provide for flexible customization of graphical user interfaces.

[0003] The evolution of the computer industry is unparalleled in its rate of growth and complexity. Personal computers, for example, which began as little more than feeble calculators with limited memory, tape-driven input and monochrome displays are now able to tackle almost any data processing task. While this meteoric increase in power was almost sufficient to satisfy the demand of application designers and end users alike, the corresponding increase in complexity created an ease-of-use problem which the industry was somewhat slower in solving. Thus, designers were faced with a new challenge: to harness this computing power in a form usable by even those with relatively little computer training to smooth the transition of other industries into a computer-based information paradigm.

[0004] As a result, in the early to mid-1980's many new I/O philosophies, such as "user friendly", "WYSIWYG" and "menu driven" came to the forefront of the industry. These concepts are particularly applicable to microcomputers, also known as personal computers, which are intended to appeal to a broad audience of computer users, including those who previously feared and mistrusted computers. An important aspect of computers which employ these concepts was, and continues to be, the interface which allows the user to input commands and data and receive results, which is commonly referred to as a graphical user interface (GUI).

[0005] The success of this type of interface is evident from the number of companies which have emulated the desktop environment. Even successful concepts, however, must continually be improved in order to keep pace with the rapid growth in this industry. The advent of multimedia, especially CD-ROM devices, has provided vast quantities of secondary storage which have been used to provide video capabilities, e.g., live animation and video clips, as regular components of application displays. With these and other new resources at their disposal, application designers and users alike, demand additional functionality and greater ease of use from the desktop environment.

[0006] To consider the challenges associated with continuing GUI design, consider as an example of a GUI which has evolved over time the Finder™ user interface and information management system (simply "Finder™ user interface" hereafter) which runs on the Apple Macintosh™ computer. The Finder™ user interface is based on the aforedescribed display principles using "windows" and "icons" to help manage computer information. The main or root window is called the "desktop" area, or more generally the primary display region. The desktop, or primary display region, is always open (displayed on the screen with its contents accessible or at least partially accessible), and takes up substantially the fall display screen area when other windows are not open. The desktop is usually visible in the background when other windows are open.

[0007] Existing inside any particular window, including the desktop itself, are other information identifiers called "icons." An icon is a screen identifier associated with a particular collection of computer information. Typically an icon may represent a "file" which is either a collection of data or a program or program segment. An icon also may represent the closed state of a window. Icons are graphic images displayed on the computer screen and usually correspond to the type of information stored within the file. Icons give the user access to the particular file represented by the graphic image when the icon is visible. The use of icons and windows is well known in the art.

[0008] The "file" is the information packet that the user wishes to utilize, create or modify; each particular file has an associated name identifying the file. Therefore, any given file may be located in the information management system by knowing a file name, an iconographic representation associated with the name, or a window locator name. All information (files) situated within a particular window are identified with that particular window's own identification location within the computer information management system. Therefore, any particular file information can be retrieved knowing its particular identification name and its window name. Accordingly, the resulting screen display utilizing the Finder™ user interface may be broken down into multiple windows and graphic icons.

[0009] Another important element of this (and other) conventional user interfaces is a screen cursor. The cursor allows direct user control over the user interface as described above. The Finder™ user interface is complemented with a "mouse" and a corresponding "pointer" which makes up the cursor control device. The user has control over the mouse, which is an electro-mechanical device that translates two-dimensional mouse movement into a two-dimensional screen position movement represented by, for example, a pointer or arrowhead. The user contacts and directs the mouse. When the mouse is moved freely on a table top, then the pointer on the screen will move in a similar and proportional manner. The mouse also contains one or more push buttons which can be used to effectuate control over the cursor pointer by selecting or deselecting specific icons or other display tools. It is said that the cursor pointer is "activated" when the mouse button is depressed and the pointer remains active until the button is released. Pointer activation may also be initiated by sequences of mouse button presses, such as a "double click" interaction which involves rapidly pressing the mouse button press twice in sequence.

[0010] Access to information in a conventional user interface system for a display management system is therefore based on windows, icons and pointer movement of the cursor. To access a file, the cursor pointer is placed on the visible icon or visible file name and the pointer is activated. A closed window may be represented by an icon or a window name. A window opens when the pointer of the cursor rests on the visible icon or visible name representing the closed state of the window and the pointer is activated. Within the open

window, files may be displayed by icon or by name. An open window, of various geometries, may be rectangular and will exist within the display area of the main viewing screen on the desktop. Multiple windows may be open at one time, typically with the most foreground window corresponding to the most recently opened window and the background windows representing those opened previously. In the organization scheme described, it is appreciated that files are nested within windows and windows can be nested within other windows; the main or root window being the desktop area, or primary display region.

[0011] During a session using a window-based information system, many windows can be open at one time with many displayed icons within. Windows may overlap and partially, or entirely, hide other windows or icons. What results is that the particular information the user wants to obtain may be hidden behind several layers of windows and may be difficult to access; when an icon is hidden by another window it is temporarily not accessible. This has been referred to in the industry as the "window overlap" problem. There are several instances where window overlap problems routinely arise in the usage of conventional user interfaces. A few of the more troublesome scenarios are described below.

[0012] In order to complete a task, often the user must access a' single icon within an open window that exists in the background, that is, covered or partially covered by other windows. The desired icon ("target" icon) within the window is no longer visible, and therefore not presently accessible. The overlapping windows or those that lay "on top of" the target window must be closed or moved away ("shuffled") so that the target window and target icon are visible and thus accessible. Window shuffling is time consuming, confusing and often very tedious for the user. If multiple routine icons need to be systematically accessed in sequence then multiple window shuffling procedures may be required.

[0013] Another window overlap problem plaguing conventional user interfaces arises when the user requires two icons to complete a task and each icon is within a different window. The resulting screen display may contain several open windows from past tasks that may clutter the screen display with unwanted information. This information may obscure the desired windows and icons. In many instances the overlapping windows are not unwanted, but hold the first of the desired icons in displayable view. In order to access the second desired icon, the user may close the overlapping window that holds the first icon, then gain access to the second desired icon. Since the previously closed window holds the first desired icon it must be opened again so that the present task can be completed. Again, this process is often time consuming and confusing for the user—especially when the hidden second icon is one that is routinely required. In this case the user is engaged in constant "window shuffling" as described above.

[0014] Not surprisingly, these types of problems have received a significant amount of attention in recent years. Several user interface products have been developed which provide different solutions to the manner in which frequently used and currently active desktop objects are handled by the GUI. For example, consider the conventional GUI depicted in FIGS. **1**(*a*) and **1**(*b*). Therein, a "Desk Drawer" concept is implemented to provide selectively hideable access to frequently used desktop objects. FIG. **1**(*a*) depicts the screen **75** having a desktop area **20** with the Desk Drawer closed, wherein only the handle **10** of the Desk Drawer is visible. An open window **60** containing several document icons **55-58** which are, therefore, accessible for operations by the user via cursor **50**. The window **60** also includes a window title field **65** and window select region **74**.

[0015] When activated, e.g., by placing cursor **50** over handle **10**, the Desk Drawer "opens" to reveal its contents. In this case, icons **41**, **42**, **51** and **59** become visible. Now that these icons are visible, they too are available for manipulation by the user via cursor **50**. Thus, the Desk Drawer concept provides a mechanism for placing frequently used icons in an out of the way, yet easily accessible location. The interested reader is directed to U.S. Pat. No. 5,657,049, entitled "Desk Drawer User Interface" for a more in depth discussion of this technique, the disclosure of which is incorporated here by reference.

[0016] Another conventional GUI, i.e., that provided with the WINDOWS 95 Operating System, tackles the problem of desktop clutter by the provision of a taskbar to organize concurrently running applications as shown in FIG. **2**. Therein, the desktop window **200** includes a plurality of icons **210** as well as the taskbar **220**. The icons **210** provide "shortcuts" to applications or documents which can be invoked, e.g., by "double-clicking" on the desired icon. The taskbar **220** identifies windows which are active including both those which are maximized and "minimized", i.e., are not currently displayed on the desktop **200**. Each such active application is represented on the taskbar **220** by a corresponding button, which typically has an iconic representation of the application as well as some descriptive text. As new applications axe launched, representative buttons will be added to the taskbar **220**, from left to right. Each existing button will be scaled in length to permit the taskbar to accommodate new buttons. To "maximize" an application residing on the taskbar **220**, the user can single click on the representative button. Another feature sometimes seen in this type of conventional GUI are application bars, e.g., appbar **230**. Appbar **230** typically includes a number of smaller buttons (relative to the length of buttons on the taskbar when only a few applications are resident there), which buttons can be depressed to launch a currently inactive application.

[0017] This conventional GUI, however, suffers from the drawbacks of having a rather rigidly structured layout (e.g., the user cannot select or organize the order of the buttons on the taskbar **220**) and from difficulties in handling the representation of a large number of applications. As more buttons are added to the taskbar **220**, each individual button becomes smaller. When, for example, between 20-30 applications have been launched and minimized, the taskbar **220** begins to add new buttons as a second layer rather than continuing the line of buttons illustrated in FIG. **2**. To reach the second layer, the user must toggle the taskbar **220**, i.e., not all of the buttons are visible simultaneously on the GUI. As the power of computers and number of interesting applications, documents and other objects increases, it is anticipated that users will wish to have ready access to a growing number of objects and, therefore, will find the approach depicted in FIG. **2** to be annoying and ineffective.

[0018] Another conventional GUI which attempts to solve this particular problem can be found in the NeXT™ Operating System. As illustrated in FIG. **3**, and further described in U.S. Pat. No. 5,146,556, entitled "System and Method for Managing Graphic Images" (the disclosure of which is also expressly incorporated here by reference), this GUI provides an application "dock" **300** including a column of icons on the

right side of the screen **310**. The dock **300** is described as providing a visible mechanism for starting applications. Icons can be added and deleted to the application dock **300** by dragging them into a desired location proximate the docking area, at which time the operating system will integrate them into the dock **300**.

[0019] Although somewhat more flexible in terms of allowing the user to organize its content than the taskbar/appbar of FIG. **2**, the application dock **300** still suffers from its limitations in terms of the number of applications which can be docked at any one time. The icons in the dock are of a fixed size and, according to the user manual, are therefore limited to a maximum of 13 which can be included in the dock at any one time.

[0020] Thus, it can be seen that there remains a need in the art to design a GUI which provides the user with a larger degree of flexibility in terms of both the layout of the tool which manages these types of frequently used objects, as well as permitting a larger number of such objects to be managed and simultaneously displayed.

SUMMARY

[0021] According to exemplary embodiments of the present invention, these and other drawbacks and difficulties of conventional GUIs are overcome by providing a simple, consolidated and easily extensible facility for handling, for example, frequently used objects. For example, user interfaces according to the present invention provide a tool (referred to herein as the "userbar") which consolidates features including: launching and managing running applications; opening and managing documents and their associated windows; accessing control strip functionality; navigation to all types of uniform resource locators (URLs); and status and notification on running processes.

[0022] As mentioned above, existing tools of this type, such as taskbars and docks, are constrained in one or more of at least two ways: having a rigidly structured layout and being limited in the number of objects that they can represent in the available screen space. With respect to layout, the userbar according to the present invention is designed so that the organization of the userbar is in the hands of the user. For example, the tiles that represent the individual items in the userbar can be reorganized at will. There is virtually no structure enforced on the user, with the exception of two "bookends" which define boundaries of the facility.

[0023] With respect to screen space, the userbar according to the present invention provides a method for scaling the entire contents of the object handling facility such that literally upwards of fifty objects (or more) can be accommodated in a single, visible structure. As the objects handled by the userbar become rather small, e.g., due to the size set by the user or due to a large number of objects being added thereto, it naturally becomes more difficult to distinguish between different tiles. Accordingly, exemplary embodiments of the present invention provide a magnification effect, also referred to herein as a fisheye effect, for browsing the scaled contents of the userbar. This mechanism allows specified contents of the userbar, in particular minimized windows, to be presented at a larger size and in greater detail and legibility than other objects in the userbar. This feature permits, among other things, the individual tiles of the tool to retain their legibility and prevents the user interaction with the tool from being compromised by the scaled contents.

BRIEF DESCRIPTION OF THE DRAWINGS

[0024] These and other objects, features and advantages of the present invention will be readily understood by those skilled in the art by reading the following detailed description in conjunction with the drawings, in which:

[0025] FIGS. **1**(*a*) and **1**(*b*) depict a first, conventional user interface;

[0026] FIG. **2** depicts a second, conventional user interface which employs a taskbar and an appbar to handle objects;

[0027] FIG. **3** depicts a third, conventional user interface which employs an application dock;

[0028] FIG. **4** is a block diagram of an exemplary system in which the present invention can be implemented;

[0029] FIG. **5** is an exemplary computer system which may contain the functional blocks of FIG. **4** and in which the present invention can be implemented;

[0030] FIG. **6** illustrates a user interface including a userbar according to an exemplary embodiment of the present invention;

[0031] FIG. **7** depicts the user interface of FIG. **6** with the cursor disposed at another location within the userbar region on the screen; and

[0032] FIGS. **8**(*a*)-**8**(*d*) describe an exemplary magnification effect mechanism according to an exemplary embodiment of the present invention.

DETAILED DESCRIPTION

[0033] In the following description, for purposes of explanation and not specific details are set forth, such as particular circuits, circuit components, techniques, etc. in order to provide a thorough understanding of the present invention. However, it will be apparent to one skilled in the art that the present invention may be practiced in other embodiments that depart from these specific details. In other instances, detailed descriptions of well-known methods, devices, and circuits are omitted so as not to obscure the description of the present invention.

[0034] Exemplary embodiments of the present invention can be implemented on an Apple Macintosh™ computer system using the Finder™ user interface. However, it will be readily appreciated by those skilled in the art that user interfaces and elements of user interfaces according to the present invention can be used in combination with any system having a processor and a display. In general, such computer systems, as illustrated in block diagram form by FIG. **4**, comprise a bus **400** for communicating information, a processor **401** coupled with the bus for processing information and instructions, a random access memory **402** coupled with the bus **400** for storing information and instructions for the processor **401**, a read only memory **403** coupled with the bus **400** for storing static information and instructions for the processor **401**, a data storage device **404** such as a magnetic disk and disk drive or CD ROM drive coupled with the bus **400** for storing information and instructions, a display device **405** coupled to the bus **400** for displaying information to the computer user, an alphanumeric input device **406** including alphanumeric and function keys coupled to the bus **400** for communicating information and command selections to the processor **401**, a cursor control device **407** coupled to the bus for communicating information and command selections to the processor **401**, and a signal generating device **408** coupled to the bus **400** for communicating command selections to the processor **401**.

[0035] The display device **405** utilized with the computer system and the present invention may be a liquid crystal device, cathode ray tube, or other display device suitable for creating images and alphanumeric characters (and ideographic character sets) recognizable to the user. The cursor control device **407** allows the computer user to dynamically signal the two dimensional movement of a visible symbol (cursor) on a display screen of the display device **405**. Many implementations of the cursor control device are known in the art including a trackball, mouse, joystick or special keys on the alphanumeric input device **406** capable of signaling movement of a given direction or manner of displacement. It is to be appreciated that the cursor also may be directed and/or activated via input from the keyboard using special keys and key sequence commands. Alternatively, the cursor may be directed and/or activated via input from a number of specially adapted cursor directing devices, including those uniquely developed for the disabled. In the discussions regarding cursor movement and/or activation within the preferred embodiment, it is to be assumed that the input cursor directing device or push button may consist any of those described above and specifically is not limited to the mouse cursor device.

[0036] FIG. **5** illustrates an exemplary computer system that in which the present invention can be implemented. It will be appreciated that this computer system is one of many computer systems that may can include the present invention. Therein, a keyboard **500** with keys **502** and keypad **504** is attached to the computer **506** along with a mouse device **508** and mouse push button(s) **510** for controlling the cursor. The mouse device **508** and push button **510** make up a cursor control device **407**. It will be appreciated that many other devices may be used as the cursor control device **407**, for instance the keyboard **500** may be substituted for the mouse device **508** and button(s) **510** as just discussed above. The computer **506** also typically contains a one or more media drives **511** (e.g., floppy disk, hard disk or CD ROM) and a display screen **512**.

[0037] Having described exemplary computer systems in which user interfaces according to the present invention can be implemented, the discussion now turns to a description of such user interfaces. According to exemplary embodiments of the present invention, a userbar is provided to the user interface which solves many of the problems described above with respect to conventional user interface tools and facilities by providing extensibility, scalability and flexibility which are lacking in prior systems.

Userbar Contents

[0038] An example of the userbar according to the present invention can be seen in the user interface of FIG. **6**. Other examples are seen in subsequent figures which will be further described below. Therein, the userbar **600** includes a number (in this example sixteen) of tiles aligned along a bottom portion of a user interface, the magnification level of which varies based on the position of the cursor **610** in a manner to be described below in the section entitled "Userbar Appearance". The contents of the userbar **600** may represent a user-selected (or alternatively, an application or operating system selected) set of "super-favorite" items, i.e., items that the user has determined deserve greater ease-of-access than their broader collection of favorite items, which could be numerous. The straightforward configuration mechanism of the userbar **600** according to the present invention facilitates frequent changes to its contents, which contents are based on a user's preferences and will, therefore, naturally vary from one user to the next. Any type of item or object may be handled by the userbar **600**, however a few examples, including applications, document files and windows will now be described in order to illustrate how exemplary userbars according to the present invention may represent and manipulate various items.

[0039] Applications can be presented on the userbar **600** by, for example, one of two methods. First, the application's icon can be added to the userbar **600** as a permanent fixture, e.g., for most frequently launched applications. Alternatively, the application may not be a permanent fixture of the userbar **600**, but may be added thereto because it is currently running. Such non-permanent applications may be represented in the userbar **600** only while the application remains running and may be removed automatically by the GUI when the application is terminated. Faceless background applications, e.g., virus protection applications, if not launched from the userbar **600**, need not be represented on the userbar **600** while they are running. Similarly, application sub-processes, such as a Finder copy, need not appear as a separate application tile on the userbar **600**.

[0040] Document files can also be placed on the userbar **600**. This includes, for example, editable, non-editable (i.e., read only) and stationary files. An application's "set" files (e.g., Location Manager sets) can also be placed on the userbar **600**, but may only be useful to the user as part of the userbar **600** if selecting one of these tiles would initiate a set change. A user may choose to place multiple copies of a document onto the userbar **600**, but the userbar **600** will preferably only include a single representation of each object in a particular state. For example, if a tile representing a folder is disposed on the userbar **600** and the user opens that folder, this can result in the file management system (e.g., Finder) opening (or navigating to) the directory represented by the folder. If the user then minimizes that file management system window, the originating userbar object is then presented on the userbar **600** as a minimized window (e.g., as a folder icon). Likewise, if a document is opened from the userbar **600** and its window is subsequently minimized by the user, it is preferable that a representation of the minimized window replace the image used for the originating document's tile, rather than adding a second tile to the userbar **600** for that object.

[0041] Document windows, when minimized by the user, are placed on the userbar **600** and remain there until either closed or maximized, which process is described in more detail below under the section entitled "Userbar Functionality". The image appearing on the tile can be, for example, either: 1) a dynamically scaled representation of the actual window contents, or 2) an appropriate image provided by, for example, the application, such as the window's proxy icon. For example, a minimized Finder window might more appropriately be presented as a folder icon, e.g., icon **620** in userbar **600**, as opposed to providing a scaled image of the Finder window's content on the userbar **600**.

[0042] In addition to applications, documents and windows, many other types of items may reside on userbar **600**. For example, any type of system-supported uniform resource locator (URL) file types can be placed on the userbar **600** including, but not limited to, file types having the extensions: http, ftp, news, mailto, at, afp and file. Additionally, developer-defined preference or setting modules (e.g., a slider control to adjust the computer's speaker volume) can be added to

the userbar **600** by the user. Adding such preference or setting modules to the userbar **600** may be accomplished by, for example, dragging pre-defined objects from the Finder to the bar.

[0043] According to exemplary embodiments of the present invention, two items are permanent fixtures of the userbar **600**. These items, denoted by reference numerals **630** and **640**, act as "bookends" that contain the contents of the userbar **600** between them. Those skilled in the art will appreciate that the selection of specific tiles to act as bookends may vary from implementation to implementation of the present invention. However, in the purely illustrative exemplary embodiments described herein the **630**, which represents the Finder process and its window list, is provided as the left bookend of the userbar **600**. This the **630** represents the running Finder process and, according to this exemplary embodiment, no other tiles may be placed on the userbar **600** to the left of this object. Similarly, a tile **640** representing the trash object acts as the right bookend of the userbar **600** in this exemplary embodiment. This the may replace any other iconic representation of the trash object on the desktop GUI. Acting as the right bookend, the user will not be able to place any other tiles on the userbar **600** to the right of this object.

[0044] In addition to the two exemplary, permanent fixtures on the userbar **600** represented in this example by tiles **630** and **640**, other predefined content may be placed on the userbar **600** at either the user's discretion or in a manner which is predefined by the GUI. Candidate items for such permanent residency on the userbar **600** include, for example, application folders, favorites, address book, clock, web browser and e-mail applications.

Userbar Appearance

[0045] Exemplary embodiments of the present invention provide various mechanisms which impact the appearance of the userbar **600** in a manner which is intended to aid in achieving the aforementioned objectives of providing a simple, consolidated and easily extensible facility for handling frequently used objects. The userbar **600** can be implemented as a single horizontal row of items, or "tiles", each of which represent a particular object or process. Any state which is supported for each object or process in the operating system should also be supported on the userbar **600** by providing a different appearance for each state. For example, a tile disposed in userbar **600** can change its appearance as it moves from one state, e.g., selected, to another state, e.g., open or offline. This allows the user to quickly recognize the current state of each of the items on the userbar **600**. Currently executing applications can be indicated by, for example, placing an LED-like indicator graphic above or below the corresponding application's tile or icon. Applications can also supply additional tile images to be substituted for, or composited on, the running application's tile in the userbar **600**. For example, an e-mail application's tile can present the number of new messages, superimposed over the application's icon.

[0046] According to exemplary embodiments of the present invention, the default position of the userbar **600** is anchored to the bottom of the main monitor and centered horizontally therein. Thus, the userbar **600** maintains a position aligned relative to the horizontal center of the screen, regardless of the number of items or tile placed in the userbar **600**. Depending upon the desired implementation, the user may or may not be permitted to relocate the userbar **600** from its default position.

[0047] In terms of the size of the userbar **600**, according to this purely illustrative exemplary embodiment, the userbar **600** has a default size of 64×64 pixels. This default height may change at locations associated with the cursor position as described below in the section entitled "Variable Magnification of Userbar Tiles". A gap of a few pixels may be provided between the bottom of the userbar **600** and the bottom of the screen to allow windows that are placed, or dragged, below the height of the bar to remain partially visible. Alternatively, the userbar may be provided at the very bottom of the display space such that no gap exists. In fact, the userbar **600** may be located in any desired space on the display.

[0048] The userbar **600** is, according to these exemplary embodiments, the topmost window on the screen, i.e., all other windows appear behind the userbar **600**. Applications, when creating or zooming document windows, should place the bottom of the document window above the top of the bar so as not to obscure any portion of the window with the userbar **600** which would otherwise overlay the newly created window.

[0049] Each tile can have a label associated therewith. For example, in FIG. **6** the label "Clock" can be seen centered above tile **610**. Those skilled in the art will appreciate that the label could alternatively be displayed below the corresponding tile. According to exemplary embodiments of the present invention, labels for each tile are only visibly displayed on the monitor while the cursor is proximate the corresponding tile. For example, as the cursor moves into a region associated with a particular tile, the label associated with that the is displayed. When the cursor moves out of this region, the tile label will vanish. Tile labels, when made visible or invisible, can do so in a manner which makes them appear to fade in or fade out. In current embodiments of the present invention, when the cursor **610** enters the userbar region, a fade in time of zero milliseconds is provided, although those skilled in the art will appreciate that this value may be changed. In fact, the fade in and fade out values mentioned herein may be user changeable, e.g., using the user preferences dialog box described below with respect to FIG. **7**.

[0050] As the cursor **610** continues to roll across tiles on the userbar **600**, the appropriate tile label fades in while the previous tile label fades out. Current embodiments of the present invention provide, however, provide a value of zero milliseconds for fading in and fading out of the tile labels as the cursor moves across the userbar **600**. If the cursor **610** leaves the userbar **600**, the last label presented persists for a short period, e.g., 250 milliseconds, then fades out. Although this exemplary embodiment of the present invention is described in the context of only displaying a single label at a time, those skilled in the art will appreciate that this feature of the present invention may be readily adapted to varying implementations. For example, the tile labels associated with the current tile over which the cursor is positioned, as well as the two tiles to either side of the current tile, could be simultaneously displayed. Again, the number of the labels being displayed may be user settable.

[0051] According to another exemplary embodiment of the present invention, in order to assist the user in managing the userbar's contents, separator tiles can be provided. These colored or translucent separator tiles can be placed between existing tiles to provide a visual distinction between groups of

tiles. For example, the separator tiles may be provided as a half-width tiles (e.g., 32 pixels) and/or quarter-width tiles (e.g., 16 pixels).

Variable Magnification of Userbar Tiles

[0052] As mentioned above, a significant benefit of the present invention may be found in the ability to permit a large number of tiles to reside in a single row of the userbar **600**. Of course, the number of tiles which can fit in the userbar **600** in a single row is dependent upon the screen size and the size of each individual tile. Since screen sizes are fixed, the only way to increase the number of tiles in the userbar **600** is to reduce their size. However, at some relatively small size, the images in each the will not be distinguishable by the user. Exemplary embodiments of the present invention address this problem by providing for variable magnification levels as between different tiles on the userbar **600**.

[0053] Referring again to the exemplary embodiment illustrated in FIG. **6**, it will be seen that a cursor **610** rests on top of one tile in the userbar **600**, i.e., the tile having the descriptive legend "Clock" thereabove. It will immediately be recognized that the "Clock" tile on which the cursor **610** rests has been magnified to be larger than the surrounding tiles in the userbar **600**. This magnification is attributable to the "fish-eye" effect which can be provided to the userbar **600** according to exemplary embodiments of the present invention.

[0054] In this particular exemplary embodiment, not only is the tile upon which cursor **610** rests magnified, but so are surrounding tiles. The amount of magnification can vary as between tiles proximate the cursor position. In this example, the magnitude of the magnification level is a function of the distance of each tile edge from the cursor **610**. Of course those skilled in the art will appreciate that there are many different types of magnification techniques and algorithms which can be employed to provide this type of functionality. For example, any number of tiles to either side of that over which the cursor is resting could experience some level of magnification. Alternatively, only the individual tile over which the cursor is resting can be magnified. The level (percentage) of magnification of each tile can also be varied in any desired manner.

[0055] At a more fundamental level, however, it will be appreciated that this magnification permits one or more selected tiles in the userbar **600**, i.e, the tile which is pointed to by the cursor **610** as well as tiles proximate to the cursor **610**, to be readily viewed and identified even while other tiles residing in the userbar **600** are possibly more difficult to distinguish. This magnification functionality, among other techniques associated with user interfaces according to the present invention, permits the userbar **600** to, on the one hand, handle many more objects than was possible in conventional user interfaces in a single, simultaneously viewable manner while, on the other hand, allowing the user to readily identify and manipulate individual objects residing on the userbar.

[0056] As the cursor **610** moves over the userbar **600**, certain tiles experience increased magnification, while other tiles' magnification decreases, based on their relative distance to the current cursor position. Compare, for example, FIG. **7** with FIG. **6**. Although these two figures depict the same userbar **600**, i.e., with the same tiles residing therein, the magnification characteristics differ. In FIG. **7**, the cursor **610** now rests over the tile entitled "8". Some of the tiles surrounding "8" are also magnified, while the "Clock" tile has returned to its unmagnified state and is now displayed at a default size setting. The magnification effect can be canceled when the cursor leaves the content region of the userbar **600**, as determined by the level of magnification being applied (which level is user-selectable as described below). For example, if the default size of the tiles is 64 pixels, but the tile on which the cursor is currently positioned has been scaled up to 128 pixels due to the magnification effect, then this effect will not end until the cursor leaves the 128 pixel top boundary of the userbar **600**.

[0057] Also seen in FIG. **7** is an exemplary userbar settings screen **620** which permits the user to vary the tile size, magnification characteristics and the show/hide characteristic of the userbar. This feature of exemplary user interfaces according to the present invention will be described in more detail below.

[0058] With reference now to FIGS. **8**(*a*)-**8**(*d*), a more detailed example of a variable magnification function according to the present invention will now be described. Therein, each tile is placed along a reference baseline **800** (which may be located in any position and with any orientation on the display). As seen in FIG. **8**(*a*), each the has the same height h, but may have different widths. FIG. **8**(*a*) shows the default display state of a group of tiles residing in userbar **600** when the cursor **610** is outside of the userbar region, i.e., the variable magnification effect has not been invoked.

[0059] To establish the variable magnification function, a scaling height H and an effect width W are chosen. Either, both or neither of these parameters may be user settable. For example, the user preferences dialog box **720** in FIG. **7** provides an example wherein the user can select the scaling height H via a slider. The effect width defines a region wherein any tile that lies even partially within W pixels to either side of the cursor position within the userbar region will be scaled. For example, as shown in FIG. **8**(*b*), the darker tiles will be scaled based on the cursor **610**'s position, while the lighter tiles on either end of the userbar **600** will not be scaled. The parameter W can be chosen to be a multiple of the default tile width, however such a relationship is not required.

[0060] After determining the effect region based on W and the cursor position, a scale amount S based on the effect width and change in height of the tiles is calculated according to equation (1).

$$S((H-h)\div 2)\div \text{sine}(\pi\times(h\div 2)\pm(W\times 2)) \quad (1)$$

Tiles outside the effect region will be shifted away from the cursor position by the scale amount S, resulting in the userbar **600**'s width increasing by up to 2S.

[0061] When the cursor **610** enters the userbar region, two distances (d_1 and d_2) are calculated for each tile. More specifically, for each tile the distance d_1 from the cursor **610**'s position to the left edge of tile and the distance d_2 from the cursor to the right edge of the tile are calculated as seen in FIG. **8**(*c*). If the value of either d_1 or d_2 lies outside the range $\{-W, W\}$, then the value is changed to be the closest of $-W$ and W. Scaled values d_1' and d_2' are then calculated using the following sine functions:

$$d_1'=S\times\text{sine}(\pi\div 2\times d_1\div W) \quad (2)$$

$$d_2'=5\times\text{sine}(\pi\div 2\times d_2\div W) \quad (3)$$

Each tile is then redrawn between d_1' and d_2' having a size which is scaled equally in both width and height from the lower left hand corner by a factor:

$$1+(d_2'-d_1')\div(d_2-d_1) \quad (4)$$

Those skilled in the art will appreciate that the foregoing is merely an illustrative example of a particular, yet still exemplary, embodiment by which a variable magnification effect

according to the present invention can be implemented. Moreover, although these exemplary embodiments describe user interfaces wherein the variable magnification effect is invoked when the cursor moves into the userbar **600** region, i.e., when the cursor crosses a border of one of the tiles residing in the userbar **600**, those skilled in the art will further appreciate that the magnification effect can also be invoked earlier, e.g., when the cursor moves to within some predetermined distance of one of the tile borders.

Userbar Functionality

[0062] Having described examples of userbar contents and appearance according to exemplary embodiments of the present invention, the discussion now turns to exemplary techniques in which the userbar **600** can be implemented to provide desirable user interface functionality. According to one exemplary embodiment of the present invention, the userbar **600** is not implemented as a container and, therefore, it cannot "hold" file system objects. Therefore, an object placed on the userbar **600** by the user can be implemented as an alias to the corresponding file system, or source object. This means that, for example, moving objects from the userbar to the trash **640** will not result in the source file being destroyed.

[0063] The userbar **600** according to the present invention has many functions and features including launching, navigation and process management. The userbar **600** can also support drag launching and storage, e.g., the ability to open or store a document by dragging it to an application or folder, respectively, that resides on the userbar **600** as a tile.

[0064] As a navigator, the userbar **600** provides a method for users to easily access, or navigate to, favorite "places", including but not limited to windows. For example, according to exemplary embodiments, all system-supported universal resource locators (URLs), as well as local or remote directories, can be placed on the userbar **600**. However in the context of windows, minimized windows can be scaled and added to the userbar **600**. Minimized windows can be presented on the userbar **600**, for example, as either thumbnails of their content or by their window proxy icon. For example, the minimized window of a Finder's view of a folder may be more appropriately shown as the folder icon rather than a thumbnail of the window's content. Applications' icons on the userbar **600** can provide a contextual menu of their open document windows, thereby allowing users to select a specific window to bring to the front.

[0065] As a process manager, the userbar **600** provides a method for users to identify and switch between running applications. For example, the userbar **600** will permit users to hide/unhide processes and perform other such actions through contextual menus or modified mouse clicks. An application's status, e.g., not running, launching, running and running but hidden, can also be indicated by the userbar **600**, e.g., by changing an appearance, behavior or other characteristic of the application's representative tile on the userbar **600**. An application can update its status on the userbar **600**, resulting in a change in the appearance or behavior of its representative tile. For example, a tile representing an e-mail application that is resident on the userbar **600** can be overlaid with a number representing the number of new messages in a user's inbox. This number can be updated and changed to reflect changes in the status of the in-box, e.g, increasing as new messages are received in the inbox or decreasing after the user reviews his or her messages.

[0066] In operation, according to this exemplary embodiment, all running applications will appear on the userbar **600**. When launched, these applications will place their tiles to the immediate right of the left bookend tile **630**. If the tile is left untouched while the application is running, then that tile will disappear from the userbar **600** once the application is quit. If the user changes the position of the application tile in the userbar **600** while the application is running, then the tile is designated as a permanent resident of the userbar **600** and does not disappear when it has finished running.

[0067] Items can be added to the userbar **600** by dragging them into the userbar's content region. During a drag, if the cursor **610** enters the region of the userbar **600**, the userbar **600** will expand, e.g., at the nearest point between two existing tiles, to accommodate the item(s) being dragged. This permits new items to be inserted at any position in the bar. While expanded, tile images of the items being dragged can be visible in the bar in the positions they would occupy if dropped within the bar. These can be displayed as translucent "insert target tiles" to provide the user with an idea of how the userbar **600** would appear after dropping the items onto the userbar **600** at that position.

[0068] Having entered the userbar **600** during a drag, if the cursor **610** continues across the userbar **600**, the insert target tile(s) move, following the horizontal position of the cursor **610**. The position of the cursor **610**, relative to the center-point of the underlying tile, is used to determine at what point the existing tile and insert target tiles swap positions. When item(s) are dropped, the actual tile image(s) representing the item(s) replace their respective translucent insert target tile (s). An animation sequence may be displayed to reinforce this action.

[0069] The user can reorder, or reposition items on the userbar **600**. As mentioned above, in this exemplary embodiment only two items, the Finder application tile **630** and the Trash tile **640** cannot be repositioned. Instead, these tiles remain as bookends, defining the boundaries of the userbar's contents. Reordering items can be implemented in a straightforward manner by dragging an item (tile) to a new position, with similar graphical feedback being provided as for the process of adding an item to the userbar **600**. That is, when removed the tile image can be changed to its translucent insert tile image and, when reinserted into the userbar **600**, the tile will reacquire the image associated with its current state. Multiple items can be repositioned at the same time if the user first selects a plurality of items in the userbar **600** prior to initiating a drag operation.

[0070] A selection of multiple tiles can be made by, for example, defining that performing a shift-click on a tile will only select that tile, with no further action occurring. If the user maintains the shift key in a depressed state, additional tiles can be selected. Subsequent dragging, initiated on one of the pre-selected tiles will affect all selected tiles. If a discontiguous selection of tiles is repositioned within the userbar **600**, the selection will become contiguous after the drop.

[0071] Items can be removed from the userbar **600** by dragging them to the trash tile **640**, or to the desktop. The trash tile **640** will provide appropriate drop feedback (e.g., sound and/or animation) when an item is dragged thereover. As mentioned above, since the userbar **600** is preferably not a container which holds original source identifiers, but instead only aliases (i.e., pointers), this operation will only delete the userbar representation of the item and not the source object in the operating system. Alternatively, as will be appreciated by

those skilled in the art, the userbar **600** could also be implemented using real file system objects rather than aliases.

[0072] The userbar **600** can be implemented to permit the user to interact with the items resident thereon through, for example, a single mouse click. According to this exemplary embodiment, single-clicking on tiles resident in userbar **600** will produce the following results:

[0073] On application tiles, this operation will launch the application.

[0074] On a document tile, this operation will open the document with the appropriate application, launching the application if necessary.

[0075] On a URL tile, this operation will cause the destination to be presented using the appropriate application.

[0076] On a control strip, the module's interface (i.e. a menu) will be presented. Clicking outside of the interface will dismiss the interface.

[0077] On a minimized window, this operation will cause the window to be maximized

The user can switch between running applications by clicking on the desired application tile. This will result in that application, and all of its associated windows, being brought forward in the window layering order.

[0078] In addition to drag and drop configuring of the userbar **600**, users can drag and drop files, and other userbar items, onto tiles which reside on the userbar **600**. For example, a document can be dragged and dropped onto a tile representing a word processing application in the userbar **600**, resulting in the word processing application being launched and then opening the dropped document. Alternatively, a file can be dropped onto a folder residing on the userbar **600**, resulting in the file being moved or copied to the target folder.

[0079] As will be appreciated by those skilled in the art, it is useful to provide a mechanism which permits the GUI to distinguish between an operation wherein a user is adding an item to the userbar **600** and an operation wherein a user is dropping an item onto an existing tile which is already resident on the userbar **600**. According to exemplary embodiments of the present invention, a modifier key, pressed anytime during the drag but prior to the drop, will force a drop action in place of the normal insert action. This modifier acts as a toggle between insert mode and drop mode and, if released during the drag, a drop will result in an default insert instead. Toggling the state of the modifier key will result in the bar opening (to accept an insert) and closing. During a modified drag, eligible target tiles can be highlighted to denote that they can receive the object type being dragged. The user can continue to drag items across the userbar **600**, effectively browsing for eligible targets.

[0080] The userbar **600** also provides a number of window management solutions that are intended to allow users to make better use of limited screen space. For example, according to exemplary embodiments of the present invention, the userbar **600** does not provide direct access to all open document windows as only minimized windows are to be placed on the userbar. Access to a document window menu associated with a tile resident on the userbar **600** can be provided in, for example, one of two ways. First, a sub-menu can be displayed relative to each application tile, the sub-menu listing all of that application's open documents. Second, a dedicated application tile can be added to the userbar **600** which tile provides a menu interface to all open document windows, grouped by their associated applications.

Userbar Customization

[0081] As shown in FIG. **7**, exemplary embodiments of the present invention provide techniques and mechanisms which permit a user to adjust the manner in which the userbar **600** is displayed on the screen. In this example, a dialog box **720** is illustrated having a number of user settable preferences available therein. These preferences permit the user to tailor the appearance and behavior of the userbar **600** to suit his or her needs.

[0082] For example, as described above the default tile size of the bar may be 64×64 pixels. The default value refers to the size of a tile in its unmagnified state. However, this default value can be changed by the user, e.g., by adjusting the slider shown in FIG. **7**, e.g., within a range of 16 to 128 pixels square. The preferences dialog box **720** may also contain a set of controls which permit the user to determine when the magnification effect will be invoked and the amount of magnification provided. In the exemplary preferences dialog box **720** illustrated in FIG. **7**, a set of controls is provided for this purpose including a radio button group that permits the user to determine when the magnification effect will be invoked and a slider that allows the user to set the maximum magnification level, e.g., that associated with the tile over which the cursor **610** is currently positioned. Of course the size of the magnified tiles should not be less than the default size of the tiles and some upper limit may be desirable for the magnification, e.g., 128 pixels.

[0083] The userbar **600** can also support the ability to be hidden offscreen. The preferences dialog box **720** can include a control, e.g., a radio button group as shown in FIG. **7**, that support, for example, three states (off, on and by hotkey) for the auto-hide feature. With auto-hide on, the userbar **600** will animate (e.g., slide) downwards, offscreen when the cursor **610** is no longer within the region of the userbar **600**. Then, display space normally occupied by the userbar **600** is reclaimed and applications can be notified of this event. New or zoomed documents will then be sized by the notified applications to make use of this additional screen space. Moving the cursor **610** to the bottom of the display will reveal the userbar **600**. If the option for hiding the userbar **600** using a hotkey is enabled, the userbar **600** can then be hidden or revealed by executing a user-definable key combination.

[0084] As mentioned above, some tiles on the userbar **600** may acquire permanent residency on the userbar **600** such that, for example, when the graphical user interface is initialized the permanent tiles are automatically displayed within the userbar's display region. The designation of tiles as permanent or non-permanent may also be controlled by the user through the preferences dialog box **720**, e.g., by using a select/deselect permanent tiles function (not shown in FIG. **7**) which permits a user to identify objects for designation as permanent and, optionally, to select an image for the corresponding tile to be located on the userbar **600**.

[0085] The above-described exemplary embodiments are intended to be illustrative in all respects, rather than restrictive, of the present invention. For example, although the foregoing, illustrative embodiments of the present invention depict a userbar as being a row of tiles disposed along the bottom of a display or screen, those skilled in the art will appreciate that userbars according to the present invention may be displayed anywhere in the display space and with any orientation therein. Userbars according to the present invention can have any desired shape, e.g., they could be nonlinear shapes, or could be presented as multiple rows of tiles. Thus

the present invention is capable of many variations in detailed implementation that can be derived from the description contained herein by a person skilled in the art. All such variations and modifications are considered to be within the scope and spirit of the present invention as defined by the following claims.

1.-**13**. (canceled)

14. A method for displaying items in a graphical user interface, comprising:

displaying a plurality of user interface items in the form of a bar, including a user interface item representing a first application that is not running; and

in response to a running of the first application, displaying an indicator graphic proximate to the user interface item.

15. The method of claim **14**, wherein the indicator graphic is above or below the user interface item.

16. The method of claim **14**, wherein the indicator graphic resembles an LED.

17. The method of claim **14**, wherein displaying the plurality of user interface items includes displaying an image or a graphic for each user interface item of the plurality of user interface items; and

wherein displaying the indicator graphic includes changing the image or the graphic for the user interface item representing the first application.

18. The method of claim **17**, wherein displaying the indicator graphic includes substituting a new image or a new graphic for the image or the graphic for the user interface item representing the first application.

19. The method of claim **17**, wherein displaying the indicator graphic includes composing a new image or a new graphic on the image or the graphic for the user interface item representing the first application.

20. The method of claim **14**, wherein displaying the indicator graphic includes displaying a numeric value associated with the first application.

21. A computer system comprising:

a display device; and

a processor configured to control said display device to display a user interface that performs the following operations:

display a plurality of user interface items in the form of a bar, including a first user interface item representing a first application that is not running; and

in response to a running of the first application, display an indicator graphic proximate to the user interface item.

22. The computer system of claim **21**, wherein the indicator graphic is above or below the user interface item.

23. The computer system of claim **21**, wherein the indicator graphic resembles an LED.

24. The computer system of claim **21**, wherein the operation to display the plurality of user interface items includes displaying an image or a graphic for each user interface item of the plurality of user interface items; and

wherein the operation to display the indicator graphic includes changing the image or the graphic for the user interface item representing the first application.

25. The computer system of claim **24**, wherein displaying the indicator graphic includes substituting a new image or a new graphic for the image or the graphic for the user interface item representing the first application.

26. The computer system of claim **24**, wherein displaying the indicator graphic includes composing a new image or a new graphic on the image or the graphic for the user interface item representing the first application.

27. The computer system of claim **21**, wherein the operation to display the indicator graphic includes displaying a numeric value associated with the first application.

28. A non-transitory computer readable medium containing instructions that, when executed by a computing device, cause the computing device to perform operations including:

displaying a plurality of user interface items in the form of a bar, including a user interface item representing a first application that is not running; and

in response to a running of the first application, displaying an indicator graphic proximate to the user interface item.

29. The non-transitory computer readable medium of claim **28**, wherein displaying includes generating the indicator graphic above or below the user interface item.

30. The non-transitory computer readable medium of claim **28**, wherein displaying the indicator graphic includes generating a graphic resembling an LED.

31. The non-transitory computer readable medium of claim **28**, wherein displaying the plurality of user interface items includes displaying an image or a graphic for each user interface item of the plurality of user interface items; and

wherein displaying the indicator graphic includes changing the image or the graphic for the user interface item representing the first application.

32. The non-transitory computer readable medium of claim **31**, wherein displaying the indicator graphic includes substituting a new image or a new graphic for the image or the graphic for the user interface item representing the first application.

33. The non-transitory computer readable medium of claim **31**, wherein displaying the indicator graphic includes composing a new image or a new graphic on the image or the graphic for the user interface item representing the first application.

34. The non-transitory computer readable medium of claim **28**, wherein displaying the indicator graphic includes displaying a numeric value associated with the first application.

35. A graphical user interface generated by a computing device for display to a user, the graphical user interface including:

a plurality of user interface items arranged to form a dynamic bar, the plurality of user interface items including a first user interface item representing a first application;

when the first application is not running, the first user interface item is displayed with a first representative graphic; and

when the first application is running, the first user interface item is displayed with a second representative graphic.

36. The graphical user interface of claim **35**, wherein the second representative graphic is substantially similar to the first representative graphic with an LED-like indicator graphic added above or below the first representative graphic.

* * * * *